

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1 1. (canceled)

1 2. (currently amended) The method of claim 4~~7~~, wherein during execution
2 the deferred servicing procedure services all pending interrupts from all of the plurality of co-
3 processors.

1 3. (currently amended) The method of claim 4~~7~~, wherein the plurality of
2 pending interrupts serviced by the deferred servicing procedure includes a second interrupt
3 generated by the one of the plurality of co-processors that generated the detected interrupt.

1 4. (currently amended) The method of claim 4~~7~~, wherein the plurality of
2 pending interrupts serviced by the deferred servicing procedure includes a second interrupt
3 generated by one of the plurality of co-processors other than the one that generated the detected
4 interrupt.

1 5. (canceled)

1 6. (currently amended) The method of claim 5~~7~~, wherein in the event that the
2 value stored in the interrupt register does not indicate an interrupt, a different one of the
3 co-processors is selected and the act of reading is repeated.

1 7. (currently amended) ~~The method of claim 5,~~ A method for servicing
2 interrupts generated by a plurality of co-processors included in a multiprocessor subsystem, the
3 method comprising the acts of:
4 in response to a detected interrupt, determining whether the detected interrupt was
5 generated by one of the plurality of co-processors of the multiprocessor subsystem; and

6 in the event that the detected interrupt was generated by one of the plurality of co-
7 processors, scheduling execution of a deferred servicing procedure,

8 wherein during execution the deferred servicing procedure services a plurality of
9 pending interrupts generated by two or more of the plurality of co-processors, including the
10 detected interrupt,

11 wherein the act of determining whether the detected interrupt was generated by
12 one of the plurality of co-processors includes the acts of:

13 selecting one of the plurality of co-processors as a current co-processor;

14 and

15 reading a value stored in an interrupt register of the current co-processor,

16 and

17 wherein the act of reading includes:

18 updating a private register mapping to enable access to the interrupt
19 register of the current co-processor,

20 wherein the private mapping is not used by the deferred servicing
21 procedure.

1 8. (currently amended) The method of claim 4~~7~~, further comprising the act of
2 disabling further interrupts from the plurality of co-processors in the event that the detected
3 interrupt was generated by one of the plurality of co-processors,

4 wherein during execution the deferred servicing procedure re-enables interrupts
5 from the plurality of co-processors.

1 9. (original) The method of claim 8, wherein the act of disabling further
2 interrupts is performed at a critical priority level.

1 10. (currently amended) The method of claim 4~~7~~, wherein the act of
2 determining whether the detected interrupt was generated by one of the plurality of co-processors
3 is performed at a critical priority level.

1 11. (original) The method of claim 10, wherein the act of scheduling
2 execution of the deferred servicing procedure is performed at the critical priority level.

1 12. (original) The method of claim 11, wherein the act of scheduling
2 execution of the deferred servicing procedure includes setting a second priority level for the
3 deferred servicing procedure, wherein the second priority level is lower than the critical priority
4 level.

1 13. (currently amended) The method of claim 47, wherein the multiprocessor
2 subsystem is a graphics processing subsystem.

1 14. (canceled)

1 15. (currently amended) The system of claim 420 wherein the interrupt
2 detection module is further configured to be activated by a central processing unit of the
3 computer system in response to an interrupt signal.

1 16. (currently amended) The system of claim 420, wherein the interrupt
2 detection module is further configured to disable further interrupts from all of the co-processors
3 in the event of an interrupt from any one of the co-processors, and wherein the servicing module
4 is further configured to re-enable further interrupts from all of the co-processors.

1 17. (currently amended) The system of claim 420, wherein the interrupt
2 detection module is further configured to operate at a critical priority level and the servicing
3 module is further configured to operate at a second priority level lower than the critical priority
4 level.

1 18. (currently amended) The system of claim 420, wherein the
2 multiprocessor subsystem is configured for graphics processing.

1 19. (canceled)

1 20. (currently amended) ~~The system of claim 19,~~ A computer system
2 comprising:
3 a multiprocessor subsystem including a plurality of co-processors for processing
4 data, wherein each of the co-processors is configured to generate interrupts; and
5 a driver module configured to control operation of the multiprocessor subsystem,
6 the driver module including:
7 a schedulable servicing module configured to detect and service all
8 pending interrupts from all of the co-processors when activated; and
9 an interrupt detection module configured to schedule the servicing module
10 for activation in the event of an interrupt from any one of the plurality of co-processors,
11 wherein each of the plurality of co-processors includes an interrupt register
12 configured to indicate an interrupt, and wherein the interrupt detection module is further
13 configured to determine whether one of the plurality of co-processors generated an interrupt by
14 accessing the respective interrupt registers of the co-processors,
15 wherein the interrupt detection module is further configured to maintain a private
16 mapping for accessing the respective interrupt registers, the private mapping being used
17 exclusively by the interrupt detection module.

1 21. (currently amended) A computer program product comprising: a computer
2 readable storage medium encoded with program code, the program code including:
3 program code for determining, in response to a detected interrupt, whether the
4 detected interrupt was generated by one of the plurality of co-processors of the multiprocessor
5 subsystem;
6 program code for scheduling a deferred servicing procedure in the event that the
7 detected interrupt was generated by one of the plurality of co-processors; and
8 program code for performing the deferred servicing procedure, wherein the
9 program code for performing the deferred servicing procedure includes program code for
10 servicing a plurality of pending interrupts from two or more of the plurality of processors,
11 including the detected interrupt,

12 wherein the program code for determining whether the detected interrupt was
13 generated by one of the plurality of co-processors includes program code for:
14 selecting one of the plurality of co-processors as a current co-processor;
15 and
16 reading a value stored in an interrupt register of the current co-processor,
17 and
18 wherein the program code for reading includes:
19 program code for updating a private register mapping to enable access to
20 the interrupt register of the current co-processor, wherein the private mapping is not used by the
21 deferred servicing procedure.

1 22. (currently amended) The method of claim 4~~7~~, further comprising:
2 determining whether interrupts from the plurality of co-processors are enabled;
3 and
4 if the interrupts from the plurality of co-processors are not enabled, exiting
5 without performing further processing.

1 23. (previously presented) The method of claim 2, wherein servicing all
2 pending interrupts from all of the plurality of co-processors comprises:
3 loading by the deferred servicing procedure a mapping to the registers of a next
4 co-processor to be serviced; and
5 servicing any and all pending interrupts stored in the registers of the next
6 co-processor.